④ 整数計画法を用いた解法

松井知己 | 東京工業大学　滝田　潤 | (株) Creator's NEXT

0-1 整数線形計画

本稿では，ナンバーリンクパズルを 0-1 整数線形計画問題として定式化し，それを解く方法について述べる．0-1 整数線形計画問題とは，以下の条件 (1) 各変数は 0 または 1 の値のみをとる (0-1 変数である)，(2) 目的関数は線形関数で表されている．(3) 制約条件は線形等式，あるいは (等号付き) 線形不等式で表されている，を満たす最適化問題である．数式を用いて表すならば，

$$\text{minimize } \mathbf{c}^{\top}\mathbf{x}$$
$$\text{subject to } A\mathbf{x} \geq \mathbf{b},\ \mathbf{x} \in \{0, 1\}^n$$

と書くことができる．ただし上記の例では，すべての制約式は (等号付き) 線形不等式で記述されている．記号 $\mathbf{x}$ は，変数をならべたベクトルであり，上記の問題ではベクトルの次元は n 次元であるとしている．最大化すべき線形関数 $\mathbf{c}^{\top}\mathbf{x}$ を目的関数と呼ぶ．subject to 以下に書かれている式を制約 (式) と呼ぶ．特に $\mathbf{x} \in \{0, 1\}^n$ は，変数ベクトル $\mathbf{x}$ の各要素が 0 または 1 のどちらかであることを表しており，0-1 制約 (式) と呼ぶ．また，0 または 1 のどちらかの値しかとらない変数を 0-1 変数と呼ぶ．

0-1 整数線形計画は，古くからその性質や解法が研究されており，近年では CPLEX や Gurobi Optimizer といった強力な商用ソルバーが開発されている．

無向グラフの導入

本章では，ナンバーリンクパズルを無向グラフ上のパスを求める問題として定式化を行う．ナンバーリンクパズルの問題例は，以下のような形式で与えられていると仮定する．問題例が設定されている盤面は，$M \times N$ のサイズの正方格子 (M 行 (段)，N 列の正方格子) であるとする．盤面上のマス目のいくつかには $\{1, 2, ..., L\}$ 中の各数字が書かれている．各数字について，それが書かれているマス目はちょうど 2 個である．このパズルの目的は，同じ数字の間を線で結ぶことであり，(1) 線を交差させたり枝分かれさせたりしてはいけない，(2) 数字の置かれたマスを通過する線を引いてはいけない，(3) 1 つのマスに 2 本以上の線を引いてはいけない，というルールを満たす必要がある．

上記の問題例に対し，各マス目の中央に頂点を置いたグラフを構築する (図 -1)．2 つのマス目が隣接するとき (隣接するマス目は，上下左右の最大 4 通りが存在)，対応する頂点対を無向枝で結ぶ．この無向グラフの頂点集合を V と書き，無向枝の集合を E と書く．明らかに，V は MN 個の頂点を持ち，E は $2MN-M-N$ 本の無向枝からなる．頂点の集合は，対応するマス目の位置を用いて，$V=\{(i, j) \mid (1, 1) \leq (i, j) \leq (M, N)\}$ と定義する．枝集合は，マス目の絵に対し垂直方向の枝からなる集合 $E_v=\{((i, j), (i+1, j)) \mid (1, 1) \leq (i, j) \leq (M-1, N)\}$ と，水平方向の枝からなる集合 $E_h=\{((i, j), (i, j+1)) \mid (1, 1) \leq (i, j) \leq (M, N-1)\}$

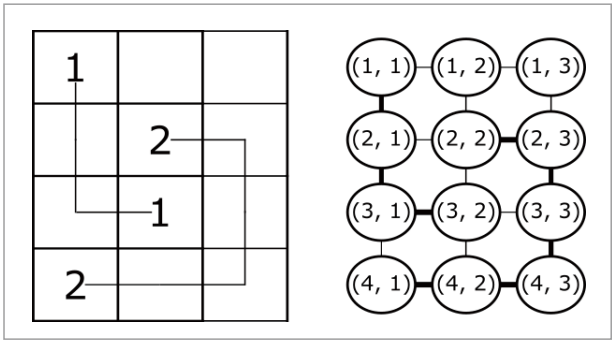

図 -1　ナンバーリンクの解を無向グラフ上で表現

を用いて $E=E_v \cup E_h$ と表す. 問題例においてマス目に書かれている数字は, 頂点に割り当てられた番号 (以下ではラベルと呼ぶ) を表す写像 $l: V \to \{1, 2, ..., L\} \cup \{\text{null}\}$ を用いて表現する. ここで null のラベルが与えられた頂点は, 対応するマス目に何も数字が書かれていないことを表すとする. 上記でも述べたように, (null 以外の) 各ラベルについて, そのラベルが割り当てられた頂点はちょうど 2 個存在する. 以上の記号を用いて, 以下本稿では, ナンバーリンクの問題例は (V, E, l) の 3 つ組で定義されているとする. ナンバーリンクの問題の目的は, V と E で定義された無向グラフ上のパス $P_1, P_2, \ldots P_L$ で以下の性質 (C1) 各パスは, 各頂点を高々 1 回通過する (この性質を満たすパスは, 初等的パスと呼ばれることが多い), (C2) 各頂点は高々 1 本のパスに含まれる (これを満たすパス集合は, 互いに点素なパス集合と呼ばれることが多い), (C3) 任意の $k \in \{1, 2, ..., L\}$ に対し, パス P_k の 2 つの端点 (パスに向きがある際は, 始点と終点と呼ばれることが多い) は, どちらもラベル k を持っている, をすべて満たすものを見つけることである. 任意のラベル $k \in \{1, 2, ..., L\}$ に対し, ラベル k を持つ 2 頂点を結ぶパス P_k を, 以下本稿ではラベル k のパスと呼ぶ.

変数と制約

前章で定義された問題例 (V, E, l) に対し, 以下ではこれを 0-1 整数線形計画問題に定式化する. 問題を記述するための 0-1 変数として

$$d_{i,j}^k = \begin{cases} 1 & (\text{ラベル}k\text{のパスが頂点}(i, j)\text{を通過}), \\ 0 & (\text{それ以外}), \end{cases}$$

を導入し, さらに頂点をパスが通過しないことを明示する変数

$$d_{i,j}^0 = \begin{cases} 1 & (\text{頂点}(i, j)\text{を通過するパスはない}), \\ 0 & (\text{それ以外}), \end{cases}$$

を導入する. 各頂点は高々 1 つのパスが通過することから, これらの変数が以下の制約

$$\sum_{k=0}^{L} d_{i,j}^k = 1 \quad \left(\forall (i,j) \in V\right) \tag{1}$$

を満たすことは明らかであろう. ラベル k のパスの両端点はラベル k を持つ頂点であることから

$$d_{i,j}^k = 1 \quad \left(\text{if } l(i,j) = k \in \{1, 2, \ldots, L\}\right) \tag{2}$$

を満たさなければならない.

問題の記述を容易にするため, 与えられたグラフの枝に対し, 新たな変数

$$v_{i,j} = \begin{cases} 1 & \left(\begin{array}{l}\text{頂点}(i,j)\text{と}(i+1,j)\text{を} \\ \text{つなぐパスが存在}\end{array}\right), \\ 0 & (\text{それ以外}), \end{cases}$$

$$h_{i,j} = \begin{cases} 1 & \left(\begin{array}{l}\text{頂点}(i,j)\text{と}(i,j+1)\text{を} \\ \text{つなぐパスが存在}\end{array}\right), \\ 0 & (\text{それ以外}), \end{cases}$$

を導入する. 各頂点について, それが $\{1, 2, ..., L\}$ 中のラベルを持つならば, その頂点の上下左右のうちいずれか 1 方向の枝が, パスに含まれることから

$$v_{i,j} + v_{i-1,j} + h_{i,j} + h_{i,j-1} = 1 \quad (\text{if } l(i,j) \neq \text{null}) \tag{3}$$

が成り立つ. 頂点 (i, j) が null のラベルを持つ場合は, その頂点を端点とする (最大) 4 本の枝のうち, $d_{i,j}^0 = 0$ ならば 2 本, $d_{i,j}^0 = 1$ ならば 0 本の枝が使われることから

$$v_{i,j} + v_{i-1,j} + h_{i,j} + h_{i,j-1} = 2(1 - d_{i,j}^0) \quad (\text{if } l(i,j) = \text{null}) \tag{4}$$

が成り立つ. ただし, 盤面の端のマス目に対応する頂点においては, その頂点を端点とする枝は 4 本存在しないこともあり, そのような場合は, 上記の制約から存在しない変数を取り除かなければならない (詳細略).

頂点の変数 $d_{i,j}^k$ と枝の変数 $v_{i,j}$, $h_{i,j}$ の整合性を保つため, 以下の制約を導入する. 任意のラベル $k \in \{1, 2, ..., L\}$ に対し, ある枝をパスが通過するな

らば，その枝の両端点を通過するパスのラベルは同じでなければならないため，以下の制約

$$
\begin{aligned}
&d^k_{i,j}+v_{i,j}-1\le d^k_{i+1,j}\le d^k_{i,j}-v_{i,j}+1 \\
&\left(\begin{array}{l}\forall(i,j)\in\{1,\ \ldots,\ M-1\}\times\{1,\ \ldots,\ N\},\\ \forall k\in\{0,\ 1,\ 2,\ \ldots,\ L\}\end{array}\right), \\
&d^k_{i,j}+h_{i,j}-1\le d^k_{i,j+1}\le d^k_{i,j}-h_{i,j}+1 \\
&\left(\begin{array}{l}\forall(i,j)\in\{1,\ \ldots,\ M\}\times\{1,\ \ldots,\ N-1\}),\\ \forall k\in\{0,\ 1,\ 2,\ \ldots,\ L\}\end{array}\right),
\end{aligned}
\tag{5}
$$

を満たさなければならない．

目的関数について簡単に記述する．たとえば，パスの長さの総和の最小化は，（どれかの）パスに含まれる頂点の総数最小化と本質的に等しいことから，$\sum_{(i,j)\in V}\sum_{k=1}^{L} d^k_{i,j}$ を最小化することによって達成することができる．実際のコンテストでは，パスが曲がることを表す変数を導入し，曲がる回数の少ない解を求める目的関数を使用したが，本稿では省略する．

上記より，ナンバーリンクパズルを 0-1 整数線形計画問題として

$$
\begin{aligned}
&\text{mininize} && \sum_{(i,j)\in V}\sum_{k=1}^{L} d^k_{i,j} \\
&\text{subject to} && (1),\ (2),\ (3),\ (4),\ (5), \\
& && \text{変数 } d^k_{i,j},\ v_{i,j},\ h_{i,j} \text{はすべて} \\
& && 0 \text{ または } 1 \text{ の値を取る},
\end{aligned}
$$

と定式化することができる．上記の定式化が正しいことを保証するには，上記の問題の制約を満たす解から，性質 (C1)(C2)(C3) を満たすパス $P_1, P_2, \ldots, P_L$ が構成できることを示す必要があるが，これについては省略する．

上記の問題を，適当なソルバー（ソフトウェア）を用いて解くことにより，ナンバーリンクパズルを解くことができる．

定式化の改善

前章において，ナンバーリンクパズルを 0-1 整数線形計画問題に定式化を行ったが，このままでは求解に時間がかかるため，さらに制約を追加することで，求解時間の短縮を図る．求解時間短縮のために追加する制約は，カットと呼ばれることが多い．

（カット A）求めるパス集合において，それらの長さの総和が最小のものが良いならば，いわゆる「コの字」のパスが存在しないことが望ましい．長さ 4 の閉路をなす枝のうち，高々 2 本のみ用いる制約として，

$$
\begin{aligned}
&v_{i,j}+v_{i,j+1}+h_{i,j}+h_{i+1,j}\le 2 \\
&(\forall\,(i,j)\in\{1,\ldots,M-1\}\times\{1,\ldots,N-1\}
\end{aligned}
\tag{6}
$$

を加える（図 -2）．

（カット B）上記（カット A）と同様の理由から，求めるパス集合において，同じラベルのパスが並走しないことが望ましい（図 -3）．たとえば頂点 (i, j) から下に伸びる枝と，頂点 $(i, j+1)$ から下に伸びる枝が両方使われているならば，それぞれを含むパスのラベルは異なるものであるべきとなる．同様の性質を右方向に伸びる枝についても導入するならば，

$$
\begin{aligned}
&v_{i,j}+v_{i,j+1}+d^k_{i,j}+d^k_{i,j+1}\le 3 \\
&\left(\begin{array}{l}\forall(i,j)\in\{1,\ \ldots,\ M-1\}\times\{1,\ \ldots,\ N-1\},\\ \forall k\in\{0,\ 1,\ 2,\ \ldots,\ L\}\end{array}\right), \\
&h_{i,j}+h_{i+1,j}+d^k_{i,j}+d^k_{i+1,j}\le 3 \\
&\left(\begin{array}{l}\forall(i,j)\in\{1,\ \ldots,\ M-1\}\times\{1,\ \ldots,\ N-1\},\\ \forall k\in\{0,\ 1,\ 2,\ \ldots,\ L\}\end{array}\right),
\end{aligned}
\tag{7}
$$

という制約が得られる．

（カット C）本コンテストの問題では，どのパスも通過しないマス目（頂点）が存在する可能性がある．

このような場合，最適解が複数存在する可能性があ

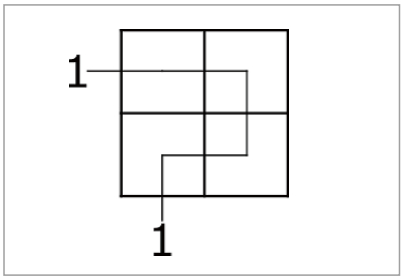

図 -2
カット A で禁止されるパスのパターン

る. 整数計画問題を解く分枝限定法では, 最適解が多数存在することは, 求解時間の増大を招くことが多い. これを避けるため,「どのパスも通らない頂点はなるべく右側へ寄せる（図 -4）」という制約を課すことによって, 求解時間の短縮を図るのが以下の制約

$$
\begin{aligned}
&d^0_{i,j} + h_{i+1,j} + v_{i,j+1} \le 2 \\
&\quad (\forall (i,j) \in \{1, \ldots, M-1\} \times \{1, \ldots, N-1\}), \\
&d^0_{i,j} + h_{i-1,j} + v_{i+1,j+1} \le 2 \\
&\quad (\forall (i,j) \in \{2, \ldots, M\} \times \{1, \ldots, N-1\}),
\end{aligned} \tag{8}
$$

である.

（カット D）上記（カット C）で述べたように, 本コンテストの問題では, どのパスも通過しない頂点が存在する可能性がある. 予備的な計算実験によって, そのような頂点の個数を固定した問題が実行不能である場合は, 比較的短い計算時間で判定できることが判明した. そこで, どのパスも通過しない頂点数を 0, 1, 2, ... に固定した問題を順に解くことで, 全体の求解時間を削減できる可能性がある. そこで, 制約として

$$
\sum_{(i,j) \in V} d^0_{i,j} = \alpha \quad (\forall (i,j) \in V) \tag{9}
$$

を導入し, 右辺の値を $\alpha = 0, 1, 2, \ldots$ と増やしながら問題を逐次解く手法を採用することができる.

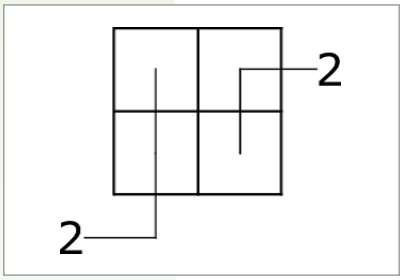

図 -3
カット B で禁止されるパスのパターン

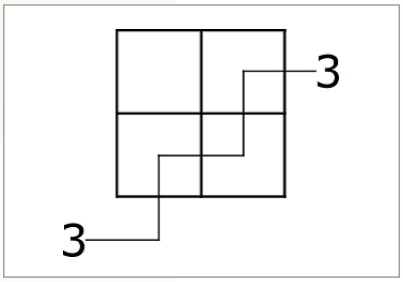

図 -4
カット C で禁止されるパスのパターン

数値実験

最後に, 数値実験を用いた定式化の性能評価について簡単にまとめる. 誌面の都合上, カット A (6) のみを追加した場合と, カット A (6) とカット D (9) の両方を追加した場合を比較する. 計算環境は, Intel Core-i7 2.8 GHz CPU, 16GB RAM という仕様を持つ計算機上で, 商用ソフトウェア IBM CPLEX 12.6.1.0 を用いた.

計算時間について表 -1 にまとめた. 表 -1 から分かるように, カット A とカット D を導入することにより, 大規模な問題でも数秒で解が得られていることが分かる. 計算実験より, どのパスも通過しない頂点の存在が, （整数計画法にとって）この問題を難しくしていると考えられる.

（2017 月 11 月 18 日受付）

表 -1 計算実験結果

問題例		計算時間（秒）	
サイズ	ラベル数	カット A (6)	カット A (6) とカット D (9)
$M \times N$	L		
10 × 10	7	0.3	0.2
10 × 10	10	0.1	0.2
10 × 18	9	0.2	0.2
10 × 18	20	0.3	0.3
14 × 18	9	2.2	0.4
17 × 15	13	2.9	0.8
16 × 19	28	2.5	1.3
18 × 18	18	2.0	0.8
18 × 21	17	44.2	0.8
18 × 21	17	> 100	3.2
35 × 19	35	> 100	6.5

■松井知己 matsui.t.af@m.titech.ac.jp

1990 年東京理科大学理工学部経営工学科助手, 1992 年東京大学工学部計数工学科講師, 2006 年中央大学理工学部情報工学科教授, 2016 年東京工業大学工学院経営工学系教授, 現在に至る.

■滝田 潤 j.takit@cnxt.jp

2012 年東京工業大学工学部社会工学科, 2015 年東京工業大学社会理工学研究科社会工学専攻修士, 2017 年（株）Creator's NEXT Web ディレクター兼数理領域アドバイザー, 現在に至る.